Chapter 25
GUI for Agent Based Modeling

Tadashi Kurata, Hiroshi Deguchi, and Manabu Ichikawa

Abstract In this paper, we discuss how to build a model by SOARS VisualShell intuitively and explain its architecture. SOARS (Agent based simulation modeling language) SOARS Project (http://www.soars.jp), Tanuma et al. (Post-proceedings of AESCS04. Springer, Japan, pp 49–56, 2004) and Tanuma and Deguchi (Inst Electron Inf Commun Eng D J90-D(9):2415–2422, 2007) is a programming language to model social phenomena by agent-based simulation. We aim to make SOARS a simulation description language by which a domain expert can simulate social interactions occurred in the real world by ones conceptual model intuitively. Therefore, a support tool for realizing and achieving specialized concepts is necessary for a domain expert to build and run a simulation model based on his/her only domain knowledge without possessing complex programming skill, and SOARS VisualShell is an application to support such intuitive modeling by SOARS.

25.1 Background

In this paper, we focus on the agent-based simulation modeling of social phenomena. By agent-based simulation, we can model a system composed of agents who make decisions autonomously, and simulate the interactions between them. On the other hand, big data analysis is becoming more important in IoT era, for experts in divergent fields, and it is becoming challenging yet promising to construct an agent-based simulation model by applying the big data [3]. We consider it important to design IoT or IoE, which handle the interrelationship among autonomous agents such as real person and things on the internet, by using agent-based simulation. As we have already implemented Pub/Sub(the standard protocol for IoT) library for SOARS, it is possible to communicate the agent of IoT through the broker [1].

T. Kurata (✉) • H. Deguchi • M. Ichikawa
Tokyo Institute of Technology, 4259 Nagatsuta-cho, Midori-ku, Yokohama, Kanagawa, 226-8503, Japan
e-mail: kurata12@cs.dis.titech.ac.jp; deguchi@dis.titech.ac.jp; ichikawa@dis.titech.ac.jp

H. Takayasu et al. (eds.), *Proceedings of the International Conference on Social Modeling and Simulation, plus Econophysics Colloquium 2014*, Springer Proceedings in Complexity, DOI 10.1007/978-3-319-20591-5_25

For agent-based simulation modeling, domain experts need to design and model the complex interactions among agents in the real world. However, it is hard work for domain experts who have no programming experiences to construct such agent-based simulation models since many such models are built by programming languages. To solve this problem, it is necessary to construct an application for experts without programming skills to build models, same as using a CAD / CAM [4, 11] based on their domain knowledge.

In order to fulfill this purpose, a Modeling GUI which could naturally realize the agent concept and their interactions that expresses is essential and necessary. Modeling GUI provides a support environment for constructing models by a domain specific language under the specific model frame. For example, Stella [10] is the domain specific language under the system dynamics model frame, and its Modeling GUI allows users to construct stock-flow type models visually and intuitively while only mathematical knowledge about the system dynamics is required. Another domain specific language is Matlab [6] whose Modeling GUI is designed to construct control models visually and intuitively under the control model frame, and only mathematical knowledge on the control theory is necessary.

However, in terms of agent-based modeling, although there are modeling GUIs which can model the agent over two-dimensional cells intuitively, i.e. NetLogo [7], few of them could express agents social interactions intuitively, and there is no Modeling GUI to express agents social role interaction neither.

25.2 Objectives

SOARS, a domain specific language under agent-based simulation model frame, was developed by Deguchi Laboratory of Tokyo Institute of Technology since 2004 [9, 12, 13] and continued to evolve. Since its simulation engine is a text-based programming language, programming skills by the text editor are required in order to construct models. As a result, domain experts without programming skills may face difficulties of constructing agent-based simulation models by using it. Therefore, SOARS VisualShell was developed as the Modeling GUI for SOARS. It enables domain experts who have few programming experiences to construct agent-based simulation models only based on their domain knowledge.

Similar to Stella, Matlab and Scratch [5, 8], SOARS VidualShell should be designed with an intuitive user interface, by achieving two objectives. The first one is to prevent the occurrence of syntactical bugs. In this way, the user could identify the semantic bug, i.e. the model design bug, when the simulation does not behave in accordance with the original intention. The other one is for domain experts to realize SOARS specific model concepts intuitively, such as social role interactions.

This paper is organized as follows. In Sect. 25.3, we will explain the design of SOARS VisualShell developed to achieve these two objectives, and conclusions and future work are discussed in Sects. 25.4 and 25.5 respectively.

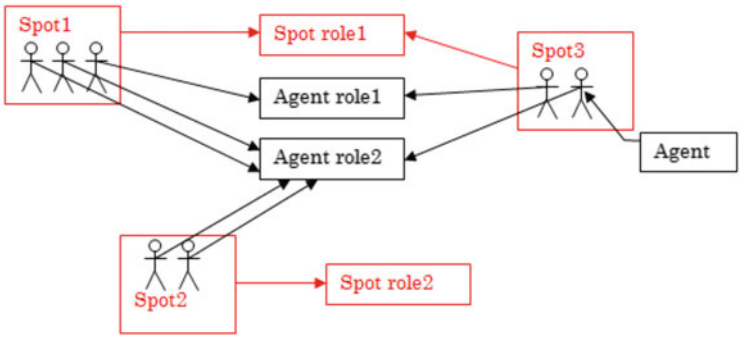

Fig. 25.1 Agent, spot and role in SOARS

25.3 Design

25.3.1 Architecture of SOARS

SOARS is composed of agents/spots as entity with their associated roles. Agents can move between any spot on the space. Agents and spots have variables, such as string, numeric, array, hash table variables and so on. A role has instructions as conditions and actions. Agent and spot can select any role to execute corresponding instructions with their variables in order to advance the simulation progresses, as shown in Fig. 25.1.

Furthermore, SOARS has the stage concept to control the instruction execution order in a role. One loop of a simulation is divided into one or more stages, which are executed in a specific order. While each stage, it is possible to define parallel executable instructions of which the execution order does not matter. By imposing such restrictions, all instructions will be executed in the correct order.

However, as its program is based on text, programming with the text editor is required, as shown in Fig. 25.2.

25.3.2 Design Concept of SOARS VisualShell

General computing languages, such as C, C++, Java, Fortran, and so on, are not domain specific languages, and the programming freedom degree is high and the description is fine-grained. As a result, their corresponding GUI is complicated to handle due to the high freedom degree, and a text editor is required for programming, such as Eclipse, Microsoft Visual Studio and so on.

On the other hand, the programming freedom of domain specific language, such as Stella, Matlab, SOARS, and so on, is low and the description is coarse-grained.

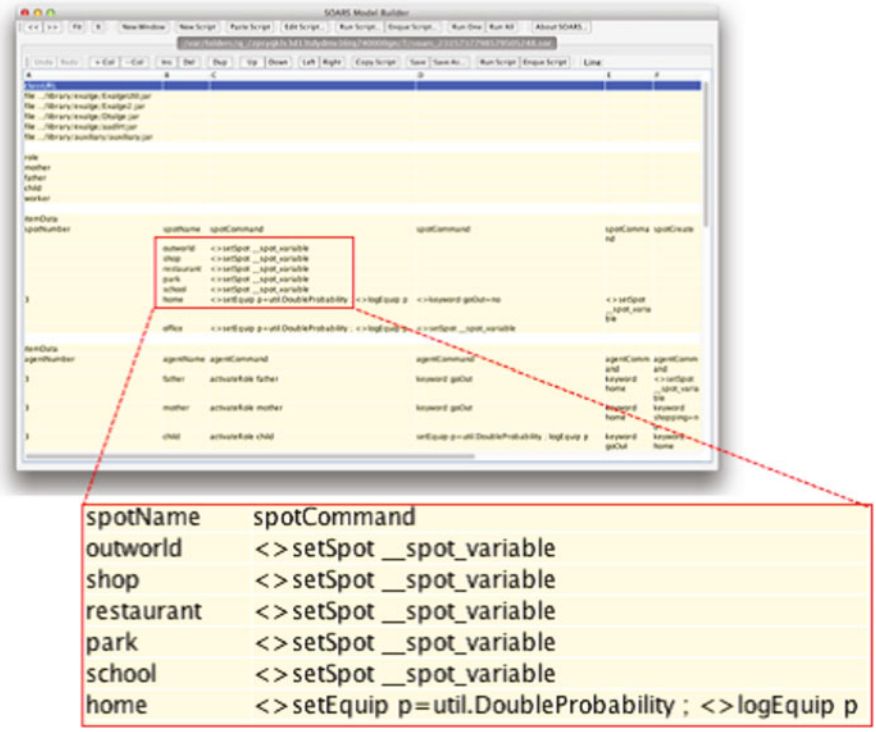

Fig. 25.2 SOARS programming in text base. It requires to type the program in each cell without syntactical error check

Table 25.1 Comparison among programming languages, model frame and modeling GUI

Programming language	DSL[a]	Model frame	Modeling GUI
General computing language[b]	No	None	None
Stella	Yes	System dynamics	Stella GUI
Matlab	Yes	Control theory	Matlab GUI
SOARS	Yes	Agent-based simulation	SOARS VisualShell

[a] Domain specific language
[b] C, C++, Java, Fortran and so on

As a result, the GUI is not complicated to handle due to the low freedom degree, and it is possible to construct the model with Modeling GUI.

The comparison among above-mentioned programming languages is shown in Table 25.1.

In this study, we construct SOARS VisualShell, the Modeling GUI for SOARS by Java.

25.3.3 User Interface of SOARS VisualShell

SOARS VisualShell is the Modeling GUI for model construction by SOARS, and can manipulate agents, spots, roles and stages of SOARS visually and intuitively. Since SOARS VisualShell can define every element to create SOARS text base program, SOARS VisualShell can describe every SOARS program as the simulation development environment. SOARS VisualShell holds the visual description language in XML format to describe elements designed for SOARS, such as agents, spots, roles and stages. There are two features of SOARS VisualShell. One is the interactive interface which does not require users to remember instructions and formats, while the other is the automatical SOARS text-based program generator.

Furthermore, SOARS VisualShell requires only mouse operations for input and output, and the keyboard input is only required to set the value of variables. Syntactical bugs are prevented as the check of input strings is done automatically. Therefore, while knowing only the agent/spot concept of SOARS, it is possible to use this GUI intuitively without holding complex programming skill. SOARS VisualShell does not require users to handle syntactical bugs. In SOARS VisualShell, agents, spots and roles are represented as icons, as shown in Fig. 25.3, and the data structure is shown in Fig. 25.4. Its user interface visualizes the structure of SOARS intuitively. Through the interface, users are able to create a new agent, spot and role only by drag and drop from the icon menu. By clicking the start button, users can create a SOARS text base program automatically and launch the simulation more easily.

There is the editing tool Dia Diagram Editor(UML editor) such as SOARS VisualShell. Dia Diagram Editor is a document generation tool to edit UML diagram [2]. On the other hand, SOARS VisualShell is a tool for automatically generating the source code of the agent-based simulation actually works. SOARS VisualShell can build the model without syntactical bugs and run it.

25.3.3.1 Agent/Spot Edit

In SOARS VisualShell, agents and spots are represented as icons. By double clicking the icon, users can define the agent/spot name, the number of agent/spot and variables, such as string, numeric, array, hash table and so on, as shown in Fig. 25.5. It is possible to specify the number of agent/spot of up to several billion. SOARS VisualShell is actually used to infection simulation model building of huge city of about 300,000.

This user interface requires only mouse operations to select a variable type, and the keyboard input is only used for setting an agent/spot definition, a variable name and the initial value of variables.

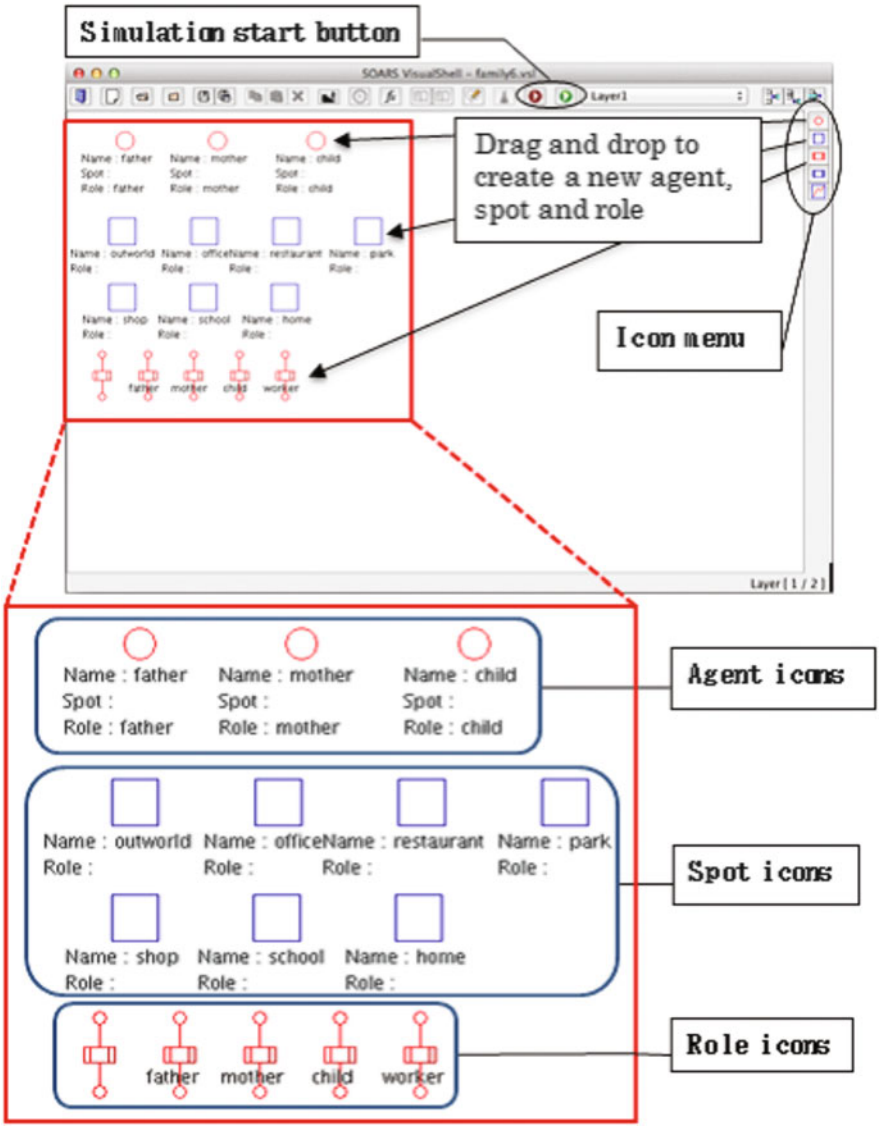

Fig. 25.3 SOARS VisualShell

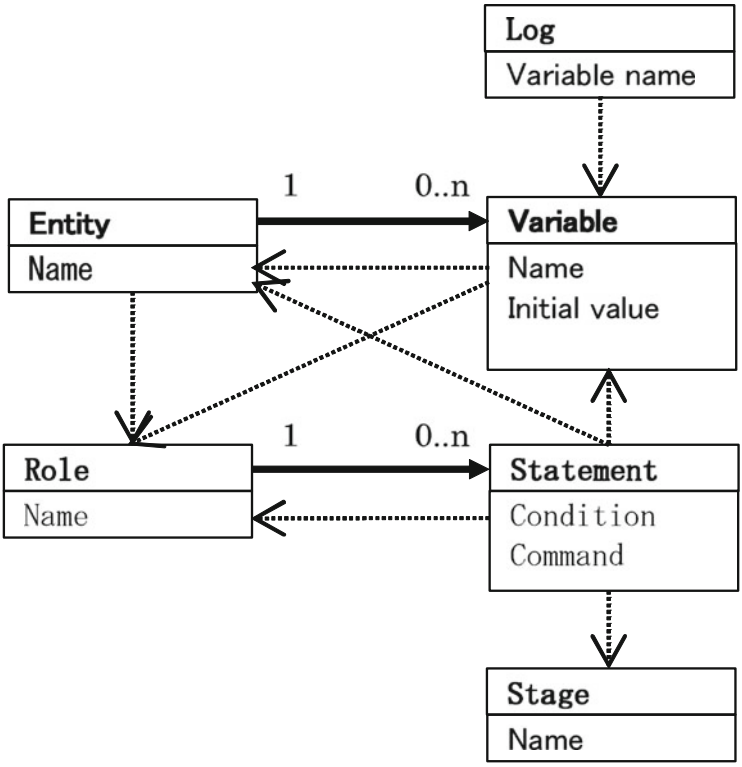

Fig. 25.4 Data structure of SOARS VisualShell

25.3.3.2 Role Edit

In SOARS VisualShell, roles are represented as icons, as shown in Fig. 25.6. By double clicking the icon, it is possible to define the role name, conditions and actions executed on each stage. The conditions and actions are defined as instructions.

This user interface is in spreadsheet format, and a stage and associated instructions should be set to each cell. Besides, it requires only mouse operations to select a stage, a condition type, an action type, an instruction and each instructions arguments, and the keyboard input is only necessary for inputting the role name.

25.3.3.3 Stage Edit

In SOARS VisualShell, stages are defined in the GUI, as shown in Fig. 25.7. This user interface requires the keyboard input for only inputting the stage name, and it requires only mouse operations to set the stage order.

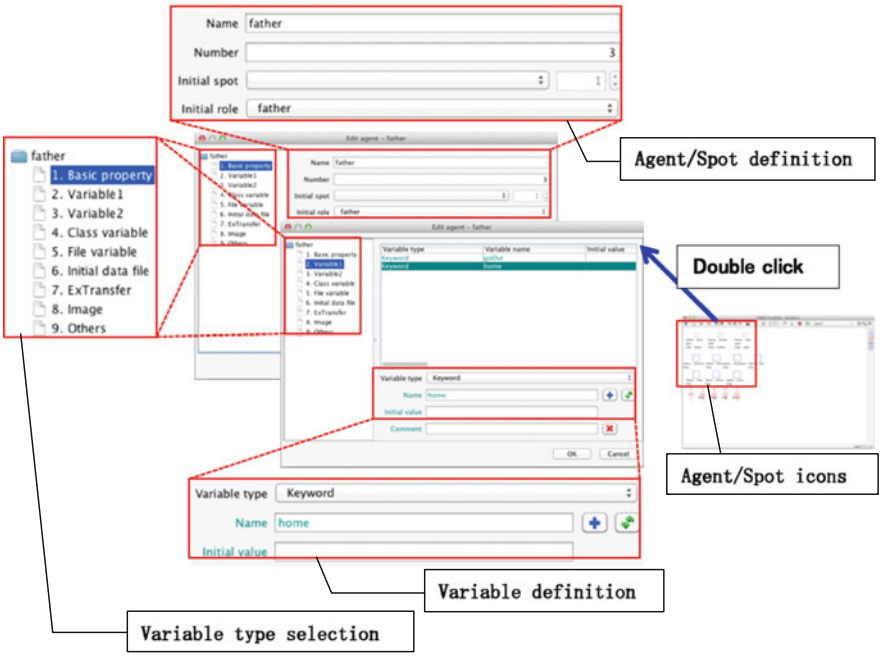

Fig. 25.5 Agent/Spot edit in SOARS VisualShell

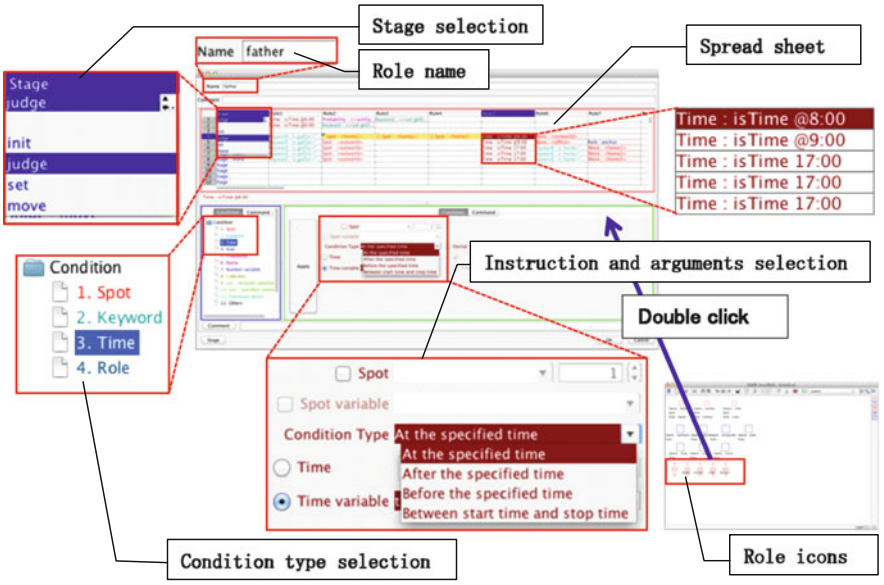

Fig. 25.6 Role edit in SOARS VisualShell

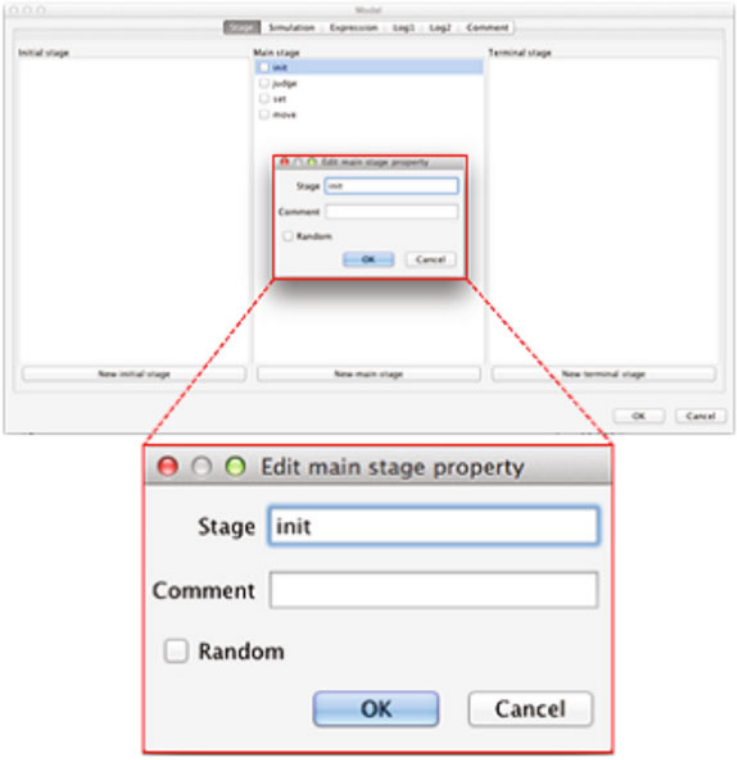

Fig. 25.7 Stage edit in SOARS VisualShell

25.3.3.4 Log Output Specification

In SOARS, the value of agent/spot variables is recorded in a log file during the simulation running. In SOARS VisualShell, the logged variables are defined in the GUI, as shown in Fig. 25.8. This user interface requires only mouse operations to select the logged variables from the list.

25.3.3.5 Simulation Condition Specification

In SOARS, it is necessary to set simulation conditions, such as the simulation start time, step time, stop time and so on. In SOARS VisualShell, they are defined in the GUI, as shown in Fig. 25.9. This user interface requires only mouse operations to select numeric values and keyboard operations for direct value input.

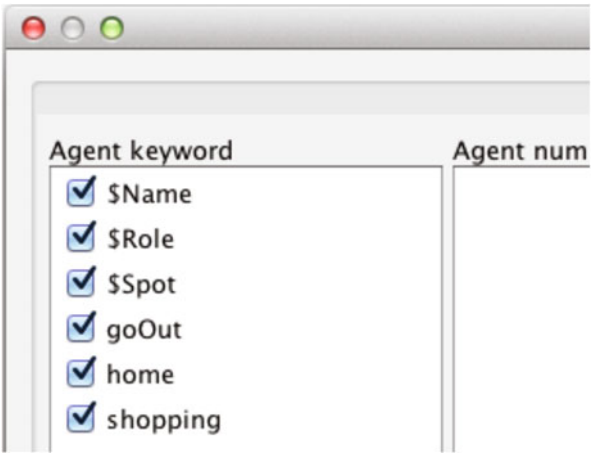

Fig. 25.8 Log output edit in SOARS VisualShell

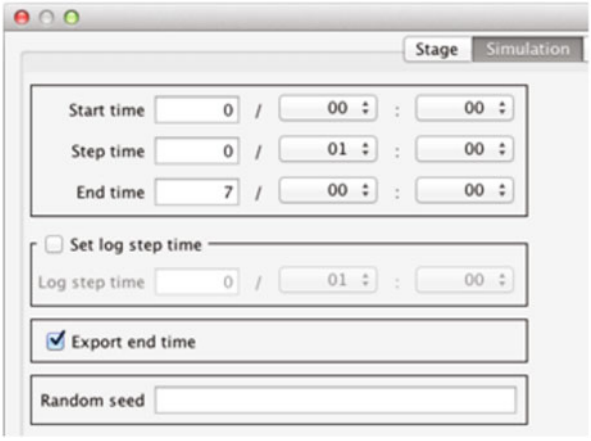

Fig. 25.9 Simulation condition edit in SOARS VisualShell

25.4 Conclusion

In SOARS VisualShell, as the concepts necessary to construct a model are expressed systematically, it is possible for the domain expert to construct a model only by selecting those elements sequentially. In other words, it is possible to construct a model merely by mouse operations. Though the string input is done via keyboard, the checking of input strings is done automatically. Therefore, it is possible to construct a model intuitively and syntactical bugs are avoided naturally. When the simulation goes against ones original intention, domain experts can identify the causes as semantic bugs, i.e. model design bugs, immediately. In addition, users can get familiar with the SOARS VisualShell operations within a very short period on any Java-installed PC (Windows, Mac, Linux).

In order to promote SOARS and for education purposes, SOARS Project holds SOARS Workshop every year [9]. By attending the intense tutorials with the

assistant of experienced SOARS programmers, anyone can construct an agent-based simulation model by SOARS through the SOARS VisualShell within a short period. SOARS VisualShell is available in the humanities universities(Waseda University, Tokyo Institute of Technology, and so on).

In this paper, we have shown that this GUI, which is designed for a domain specific language of agent-based modeling, SOARS, is effective in modeling. More specifically, since the syntactical bug never occurs by using this GUI, users can devote their effort to resolving the semantic bugs only. In addition the user can manipulate this GUI to realize the model concept intuitively while programming skills are not required.

25.5 Future Work

Agent-based modeling is expected to develop systems composed of agents in the real world in IOT era. In future, SOARS is not only expected to model autonomous agents abstracted from the real world, but also to become an agent-based designing language which could utilize big data collected from sensors and enable motion control of actuator from the real world more intuitively [1]. In addition, SOARS VisualShell is expected to evolve to enable visual description of models.

References

1. Deguchi H (2014) Multiagent simulation: 2. Social and organizational ICT architecture design in the IoE Era—from social simulation to real world OS -. Inf Process Society of Japan 55(6):589–548 (in Japanese)
2. Dia Diagram Editor (2004) http://sourceforge.net/projects/dia-installer/
3. Karnouskos S, Tariq MMJ, (2009) Using multi-agent systems to simulate dynamic infrastructures populated with large numbers of web service enabled devices. In: International symposium on autonomous decentralized systems, 2009 (ISADS 2009), pp 23–25
4. Kirk JA, Anand DK, Anjanappa M, Uppal R (1986) Implementation of a flexible manufacturing protocol. In: Proceedings of 2nd IASTED international conference. Los Angeles, CA, USA, p 71
5. Maloney J et al (2010) The scratch programming language and environment. ACM Trans Comput Educ 10(4), Article 16 (11/2010):15
6. Matlab (late 1970s) http://www.mathworks.com/
7. NetLogo (1999) https://ccl.northwestern.edu/netlogo/
8. Resnick M et al (2009) Scratch: programming for all. Commun ACM 52(11):60–67
9. SOARS Project (2004) http://www.soars.jp
10. Stella (1987) http://www.iseesystems.com/

11. Sutherland I (1963) Sketchpad: a man-machine graphical communication system. In: Proceedings of the 1963 spring joint computer conference. Spartan Books, Baltimore, MD, pp 45–53
12. Tanuma H, Deguchi H (2007) Development of agent-based simulation language: SOARS. Inst Electron Inf Commun Eng D J90-D(9):2415–2422
13. Tanuma H, Deguchi H, Shimizu T (2004) SOARS: spot oriented agent role simulator: design and implementation. In: Post-proceedings of AESCS04. Springer, Japan, pp 49–56